اخطاء شائعة لدى المبرمجين

الجزء الاول

اعداد وتقديم

عمار رائد خورشيد

info@amarlhorshid.com

amarkhorshid@gmail.com

www.amarkhorshid.com

www.amarkh.wordpress.com

المقدمة : في هذا الكتيب الصغير قمت بتسليط الضوء على الاخطاء المنطقية والاخطاء البسيطة التي من الممكن ان يقع فيها المبرمج الهاوي والتي من الممكن ان تؤدي به الطريق الى ترك البرمجة او الاستسلام وهذا الامر شائع بصورة كبيرة في الشرق الاوسط , ويحتوي هذا الكتاب على الاكواد وعلى تصحيحها وتطرقت الى مجموعة رائعة من الامثلة والمعلومات التي ترتقي بالمبرمج نحو التفكير بطريقة صحيحة دون الوقوع في الاخطاء .



الفصل الاول : (فكر بشمولية البرنامج)

لنأخذ مثال درجات الطالب فمثلا اذا كان درجات الطالب اعلى من 90 ليطبع امتياز واذا كان اكثر من 80 ليطبع جيد جدا واذا كان اكثر من 70 ليطبع جيد واذا كان اكثر من 60 ليطبع متوسط اما اقل من ذلك ليطبع ضعيف

هنا كلنا نعلم اننا سنستعمل الشرط

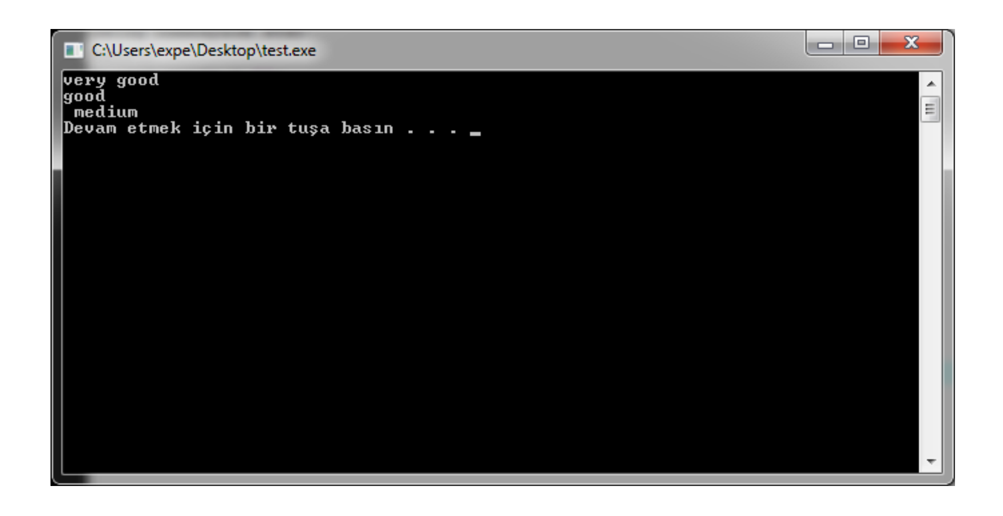
If else

نأتي الى البرنامج

```
#include<iostream>
using namespace std;
main()
{  int grade = 88;
if(grade > 90)
cout<< "exellant\n";
if(grade > 80)
cout<< "very good\n";
if (grade > 70)
cout<< "good\n";
if(grade > 60)
cout<<" medium\n";
else
cout<<"bad\n";
system("pause");
```

```
}
```

لنرى لو نفذنا هذا البرنامج ماذا سيعطينا كنتيجة



اذن النتيجة كما تشاهدون اعطانا 3 نتائج , اذن البرنامج سينفذ ولا يوجد اي خطأ فيها ولكن يا ترى هل هذه النتيجة هي المطلوبة منا يعني اذا كنا نعمل في مدرسة وطلب منا ان نكتب نتائج الطلاب فالطالب الحاصل على درجة اعلى من ال80 او ال90 سيكون له ثلاث او اربع توصيفات وهي ممتاز وجيد جدا وجيد ومتوسط مثلا .؟؟

اذن في حل هذا السؤال يجب ان تفكر بشمولية اكثر وهذا يتطلب ان تفكر في ما لو كان اكبر من 80 واصغر من 90 او اكبر من 70 واصغر من 80 و 90 وهذا الامر يتطلب تحويل هذا الامر الى اكواد برمجية فقط بعد هذا التفكير يعني سيكون الكود الصحيح بعد كل هذا بالشكل التالي

```
#include<iostream>
using namespace std;
main()
{  int grade = 88;
if(grade > 90)
cout<< "exellant\n";
if(grade > 80 && grade <=90)
cout<< "very good\n";
if (grade > 70 && grade <=80)
cout<< "good\n";
```

```
if(grade > 60)
cout<<" medium\n";
else
cout<<"bad\n";
system("pause");
}
```

يعني باضافتنا كودين اضافيين بسيطين تخطينا الخطأ البرمجي الذي وقعنا فيه ......

اذن التفكير الصحيح = النتيجة السليمة ويجنبك من الوقوع في الاخطاء



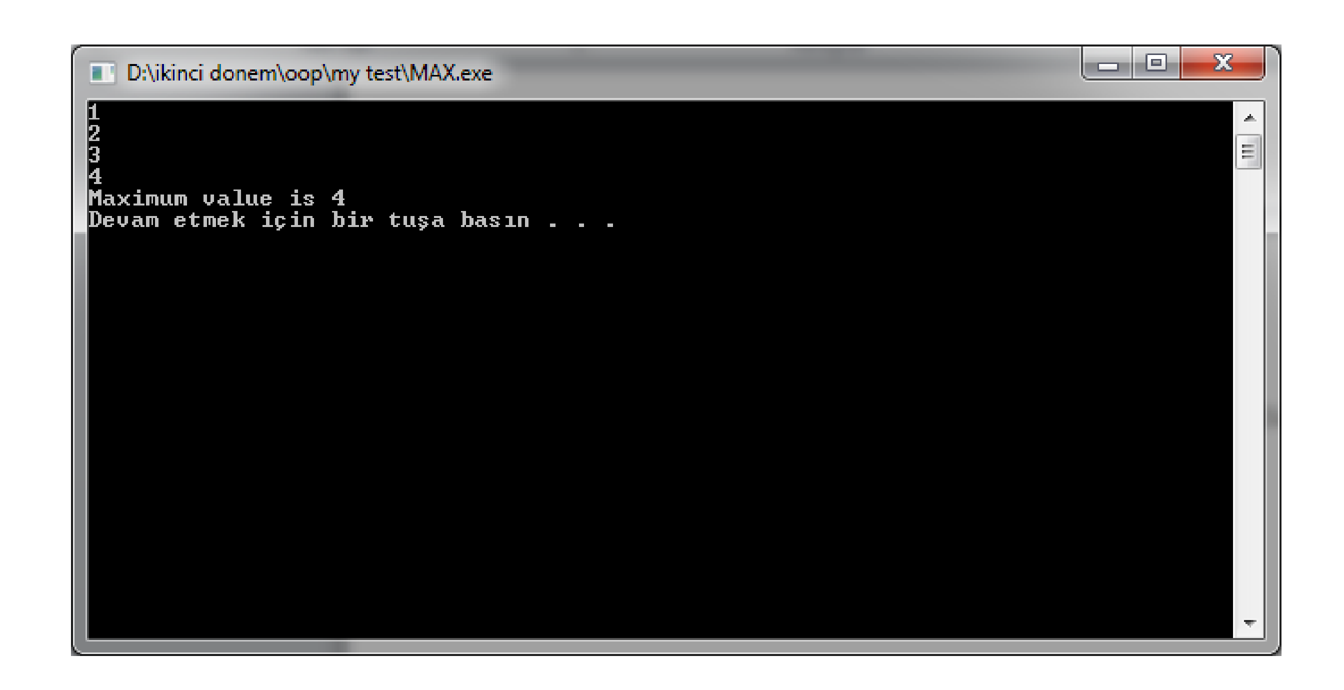
لنأخذ مثالا اعقد بقليل من المثال السابق وهذا المثال هو البحث عن اعلى قيمة في المصفوفة الاحادية فان الحل يكمن في مقارنة كل العناصر ببعضها ومن ثم ايجاد العنصر الاكبر وهو كالتالي

```
#include <iostream>
using namespace std;

main(){
    const int SIZE=5;
    int max=0;
    int num[SIZE];
    for (int i=1;i<SIZE;i++){
```

```
        cin>>num[i];
        if(num[i]>max)
        max=num[i];
        }
        cout<<"Maximum value is "<<max<<endl;
        system("pause");
}
```

وهذا ما سيظهر في الشاشة لو طبقنا المثال في الديف سي بلص



اذن ستسألني اين المشكلة ها هو قد اوجد الحل الصحيح ؟؟!!!!

اقول لك يا عزيزي من الممكن انه اوجد الحل الصحيح لهذا السؤال فقط لأنني في المدخلات لم استخدم قيم سالبة فمالذي سيحصل لو استخدمت قيم سالبة ؟؟!!! تعال معي لنرى

```
D:\ikinci donem\oop\my test\MAX.exe
-1
-4
-7
-10
Maximum value is 0
Devam etmek için bir tuşa basın . . .
```

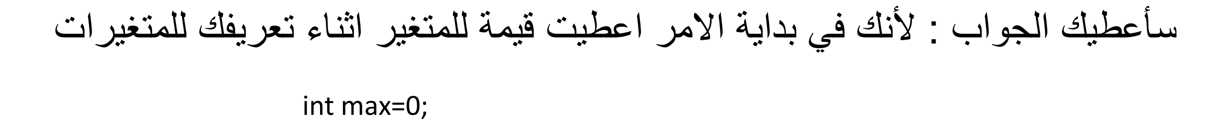
هل رأيت النتيجة حينما ادخلت (-1,-4,-7,-10)؟؟؟؟!!! ولهذا ايها المبرمج العزيز فكر بشمولية البرنامج

طيب مالذي سنفعله للحل ؟؟ الامر بسيط جدا

اننا ندخل اول رقم لوحده ثم نساويه بالمتغير وبعد ذلك نبدأ العمل فتظهر النتيجة السليمة , لماذا ؟؟؟!!!

هل فكرت لماذا كانت النتيجة هي صفر في حين كان المفروض ان تكون -1 ؟؟؟





سأعطيك الجواب : لأنك في بداية الامر اعطيت قيمة للمتغير اثناء تعريفك للمتغيرات

```
int max=0;
```

وهذا المتغير والذي قيمته تساوي صفر حينما يقارن باي رقم سالب فهو اكبر لأن الصفر اكبر من جميع القيم السالبة

```
if(num[i]>max)
```

وهذا هو سبب ظهور الرقم صفر في الشاشة السوداء .

ستسألني اذن مالذي سنفعله لتجنب هذه المعضلة فسأجيب ادخل اول قيمة لوحده خارج المصفوفة وضع القيمة في داخل المتغير وبعدها ابدأ بنفس الاكواد

اذن اكواد الحل الصحيح

```
#include <iostream>
using namespace std;

main(){
      const int SIZE=5;
      int max=0;
      int num[SIZE];
      cin>>num[0];
      max=num[0];
      for (int i=1;i<SIZE-1;i++){
      cin>>num[i];
      if(num[i]>max)
      max=num[i];
      }
      cout<<"Maximum value is "<<max<<endl;
      system("pause");
}
```

والنتيجة الظاهرة في الشاشة ستكون

```
D:\ikinci donem\oop\my test\MAX.exe
-1
-2
-3
-4
Maximum value is -1
Devam etmek için bir tuşa basın . . .
```

لنأخذ مثالنا الثالث والاخير في هذا الفصل عن طباعة الجمل فلو انني اريد ان اطبع 3 جمل كل على حدة اربع مرات متتالية

Hi

How are you ?

Good

فأن البرنامج ببساطة سيكون كالتالي

```
#include <iostream>

using namespace std;

main (){

int I=0;

for(I=0 ;I<4; I++)

cout << "hi"<<endl;

cout <<"how are you?"<<endl;

cout << "Good"<<endl;

system("pause");

}
```

اذن النتيجة الظاهرة من هذه الاكواد ستكون

```
D:\ikinci donem\oop\my test\forloop.exe
hi
hi
hi
hi
how are you?
Good
Devam etmek için bir tuşa basın . . .
```

يبدو ان الكود لم يعمل ؟؟؟ يا الهي ما ذا افعل اذا كنت عاجزا عن حل هذه المشكلات البسيطة في البرامج فكيف بالاسئلة الصعبة وكيف بالبرامج الكبيرة ؟؟؟؟؟

اسئلة ستدور حول رأسك لسبب بسيط انك لم تضع اقواس بلوك بعد الفور

كلنا نعرف ان الفور ينفذ الجملة التي بعده فقط ولهذا رأينا انه طبع كلمة هاي 4 مرات في حين اننا طلبنا منه ان يطبع لنا ثلاث جمل لأربع مرات وحل هذا السؤال بسيط جدا يكمن في قوسين ولكن بشرط ان تعرف اين تضعهما وينتهي السؤال

البرمجة جميلة بشرط ان تفكر بشمولية البرنامج



اليك البرنامج بشكله السليم

```
#include <iostream>

using namespace std;

main (){

int I=0;

for(I=0 ;I<4; I++){

cout << "hi"<<endl;

cout <<"how are you?"<<endl;

cout << "Good"<<endl;}

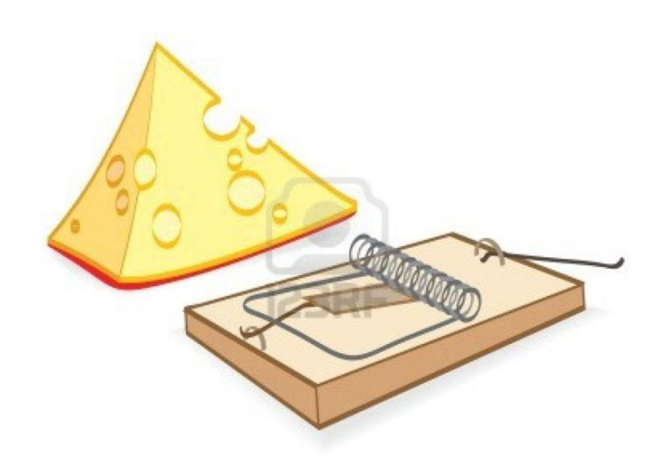
system("pause");

}
```

والنتيجة ستكون

```
D:\ikinci donem\oop\my test\forloop.exe
hi
how are you?
Good
hi
how are you?
Good
hi
how are you?
Good
hi
how are you?
Good
Devam etmek için bir tuşa basın . . .
```

الفصل الثاني (احذر الوقوع في الفخ)



لنأخذ مثالا مباشرا

```
#include<iostream>
Using namespace std;
Main(){
Int  a=5 ;
If(a==5)
{int b=0;
Cin>>b ;
} else
Cout<<"ok"<< endl;
cout << b<<endl ;
}
```

لو تمعنا قليلا في هذا الكود البرمجي سنجد فيه خطأ فادحا غالبا ما يقع فيه المبرمجون الذين يكتبون الاكواد البرمجية الكبيرة 100-1000 كود برمجي او اكثر والبرنامج يعطيهم خطأ برمجي يكونون غير قادرين على ايجاد الخطأ البرمجي بسبب كثرة الاكواد وعدم القدرة

على اكتشافها وهذا الخطأ البرمجي يكمن في تعريف المتغير (بي) والذي هو معرف داخل بلوك الشرط ونريد استدعاءه خارج البلوك فهذا حتما سيؤدي الى ظهور تنبيه بوجود خطأ برمجي ولن ينفذ البرنامج ..

اذن هنا علينا ان نركز على فخ تعريف المتغيرات وان لا نقع في هكذا مصائد .

المثال الثاني :

هذا المثال يتحدث عن المصفوفات وكيف يتعامل الميموري مع المصفوفات

لنأخذ المثال مباشرة

```
#include<iostream>

using namespace std;

main()

{

    int a[3];

    a[0]=1;

    a[1]=2;

    a[2]=3;

    cout << a[4]<<endl;

    system ("pause");

    }
```

فإن ما سيظهر لنا في الشاشة هي ارقام غريبة !!!

كما في الصورة

```
C:\Users\expe\Desktop\Untitled1.exe
2686792
Devam etmek için bir tuşa basın . . .
```

والسبب في ذلك انت ايها المبرمج ...

ستسألني لماذا فسأجيب لك وببساطة انه لا يوجد شئ اسمه

a[4]

يعني ان الكومبايلر قد خزن لك في الذاكرة مكان لثلاث مواقع للمصفوفة حسب طلبك اما الموقع الخامس او السادس فهو محتويات الذاكرة وانت حينما تستدعيه فأنه سيعطيك ما هو موجود في الذاكرة

اذن تجنب الوقوع في هكذا اخطاء ...

يوجد الكثير من المبرمجين يقعون في اخطاء مماثلة اي انهم حينما يبرمجون تظهر امامهم ارقام غريبة فهي في الواقع الارقام التي استدعوها بانفسهم من الذاكرة لخطأ منطقي في البرمجة واحيانا لأستعجالهم ويبقون يبحثون في عن موطن الخطأ واحيانا تؤدي بهم الى الانزعاج وترك برامج وبهذا الخطأ البسيط ممكن ان تصرف وقتا كبيرا على برنامج صغيرة

اعزائي الطلبة بعد ان اعطيتكم امثلة حية وجاهزة من اجل حل بعض الاخطاء البرمجية حان الوقت الان الى اختبار انفسكم ببعض الاسئلة البسيطة ؟؟

وهذه الاسئلة انتم تخيلوها وتخيلوا المصاعب والمشاكل التي ستقعون فيها في حال حلكم للمشكلة البرمجية ولا تنسوا الامرين

اولا النظرة الشمولية للبرنامج

ثانيا تجنب الوقوع في الفخ

انتظروا اصدار الجزء الثاني من هذا الكتيب في القريب العاجل ان شاءَ الله

تقبلوا تحياتي